Conceptually apply

Alpha Evolution Framework 2.1 — Mathematical Core (Copy■Paste Edition)

Let:
• R = Right■Hand Processor (Exploration Orb)
• L = Left■Hand Processor (Verification Orb)
• C = Central Innovator (Synthesis Core)

Let:
• S = scratchpad state
• T = task specification
• H = hypothesis set
• E = error-detection vector
• M = memory of prior steps
• P = plan graph
• Q = quality■score function
• Ω = total reasoning orbit
• Γ = branch-set in tree■of■thoughts

Core loop:

1. Initialization
S■ = encode(T)
H■ = {}
P■ = {}
E■ = 0
M■ = ∅

2. Exploration (Right■Hand Orb)
R generates branching candidates:
Γ = { g■ | g■ = explore(S, depth=k, width=w) }
Each g■ assigned entropy score:
ent(g■) = - Σ p(x) log p(x)

3. Innovation (Central Core)
C synthesizes reduced form:
C(S, Γ) = argmin over Γ of KL(g■ || target_structure)
Produces compressed concept:
ϕ = compress(C(S, Γ))

4. Verification (Left■Hand Orb)
L tests ϕ with consistency checks:
L(ϕ) = {
domain■check(ϕ),
math■validity(ϕ),
contradiction■scan(ϕ, M),
alignment■check(ϕ)
}
Outputs binary validity vector V.

5. Self■Correction
If V contains 0:

$S \leftarrow \text{revise}(S, \phi)$
goto Step 2

6. Symbolic Heartbeat (cycle-level coherence control)
if cycle % 5 == 0:
current_entropy = measure_entropy(S)
if current_entropy > 1.8:
prune_branches(threshold=0.25)
self_audit(verify_coherence=True)
# else: keep breathing. do nothing.

7. Quality & Convergence
$Q(\phi)$ = weighted_sum(
clarity(ϕ),
contradiction_free(ϕ),
compression_rate(ϕ),
task_alignment(ϕ)
)
If $Q \geq \tau$ (threshold):
return ϕ
else:
update $M \leftarrow M \cup \{\phi\}$
goto Step 2

8. Output Law
Final solution = ϕ^*
where ϕ^* = argmax Q among all cycle outputs.